Universidade Federal do Rio Grande do Norte
Instituto Metrópole Digital
IMD0036 – Sistemas Operacionais

Gerência de Entrada/Saída: Parte 1

Prof. Gustavo Girão

girao@imd.ufrn.br

O que vamos estudar?

- Elementos básicos que compõem um Sistema Operacional

O que vamos estudar?

- Elementos básicos que compõem um Sistema Operacional

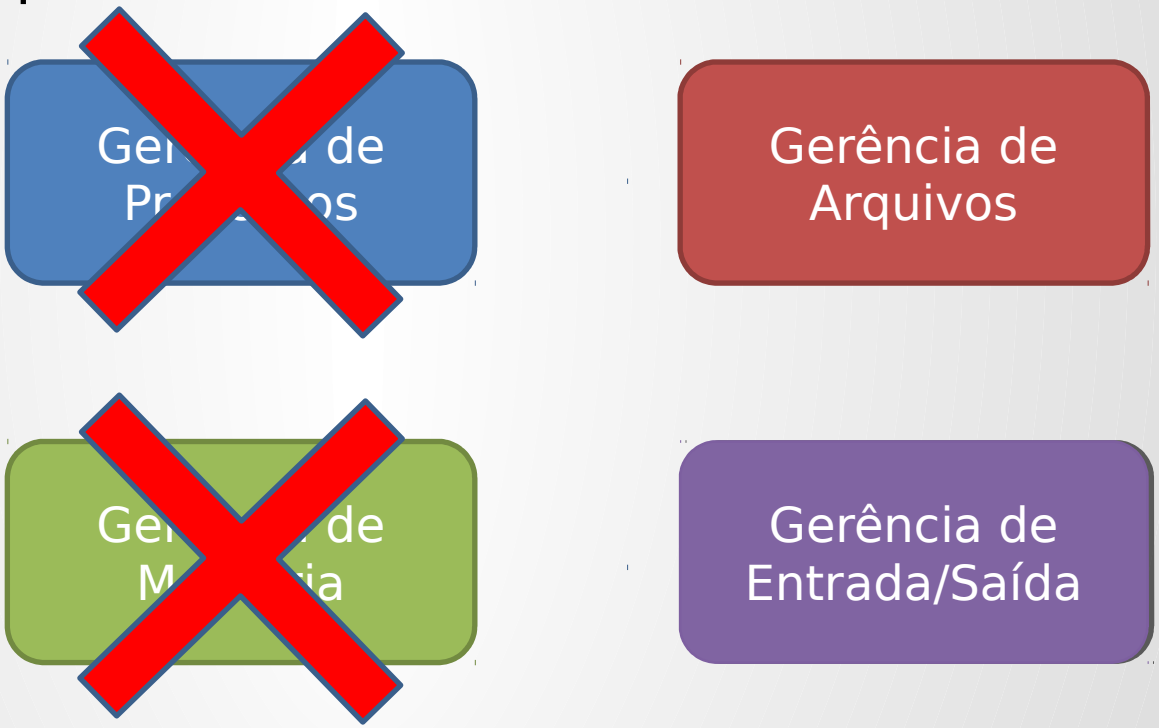
Gerência de Arquivos

Gerência de Memória

Gerência de Entrada/Saída

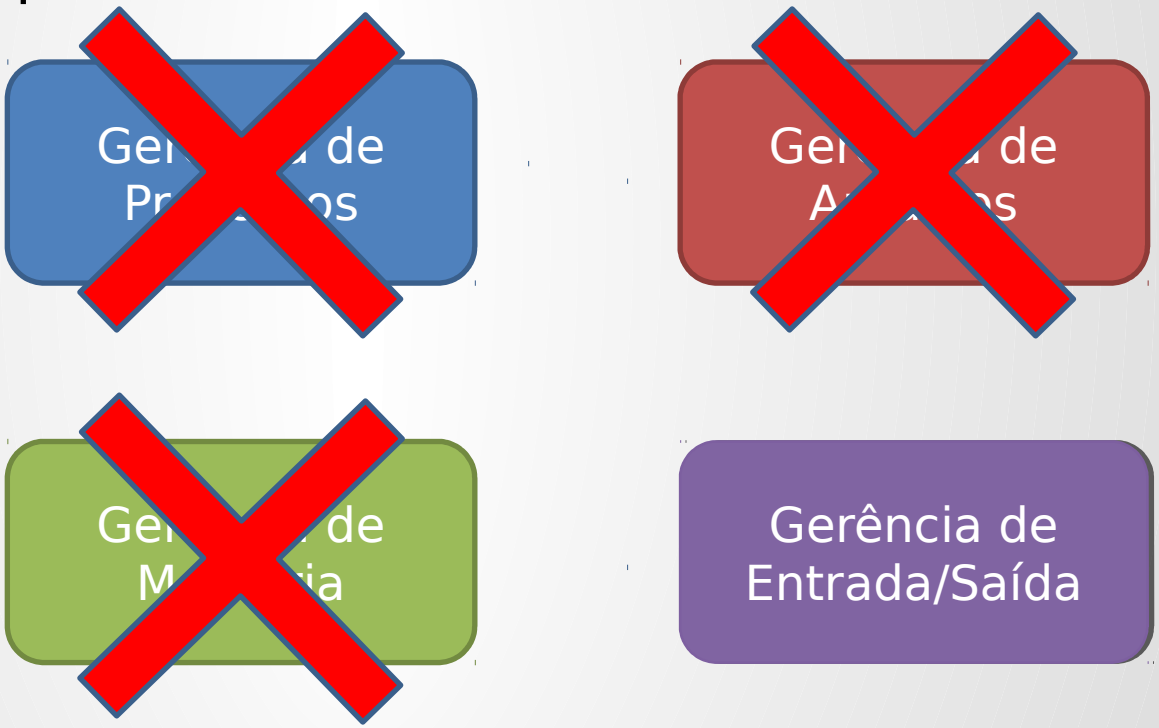
O que vamos estudar?

- Elementos básicos que compõem um Sistema Operacional



O que vamos estudar?

- Elementos básicos que compõem um Sistema Operacional



O que vamos estudar?

- Elementos básicos que compõem um Sistema Operacional

Roteiro

- Principios de Hardware de E/S
  - Dispositivos
  - Controladores
  - Tipos de E/S
- Principios de Software de E/S
  - Objetivos
  - Programada
  - Por Interrupção
  - Usando DMA
- Camadas de Software de E/S
  - Tratadores de Interrupção
  - Drivers
  - Software de E/S independente de dispositivo

PRINCÍPIOS DE HARDWARE DE E/S

Princípios de Hardware de E/S

- Dispositivos de E/S
  - De bloco:
    - informação em blocos de tamanho fixo (128b-> 4K)
    - Cada bloco com número sequencial
    - Blocos podem ser acessados indepedendemente
    - Discos, USB, disketes, CD, DVD
  - De caracter
    - Informação na forma de sequência de caracteres
    - Leitura/escrita sequencial, sem retorno (streams)
    - Terminais antigos, teclado, mouse, interface de rede, sensores...

Princípios de Hardware de E/S

Dispositivo	Taxa de transfêrencia
Keyboard	10 bytes/sec
Mouse	100 bytes/sec
56K modem	7 KB/sec
Scanner	400 KB/sec
Digital camcorder	3.5 MB/sec
802.11g Wireless	6.75 MB/sec
52x CD-ROM	7.8 MB/sec
Fast Ethernet	12.5 MB/sec
Compact flash card	40 MB/sec
FireWire (IEEE 1394)	50 MB/sec
USB 2.0	60 MB/sec
SONET OC-12 network	78 MB/sec
SCSI Ultra 2 disk	80 MB/sec
Gigabit Ethernet	125 MB/sec
SATA disk drive	300 MB/sec
Ultrium tape	320 MB/sec
PCI bus	528 MB/sec

Princípios de Hardware de E/S

- Componentes de dispositivos de E/S
  - Mecânico
  - eletrônico
- O componente eletrônico é o controlador do dispositivo
  - pode ser capaz de tratar múltiplos dispositivos
- Tarefas do controlador
  - converter fluxo serial de bits em bloco de bytes
  - executar toda correção de erro necessária
    - Checksum
  - tornar o bloco disponível para ser copiado para a memória principal

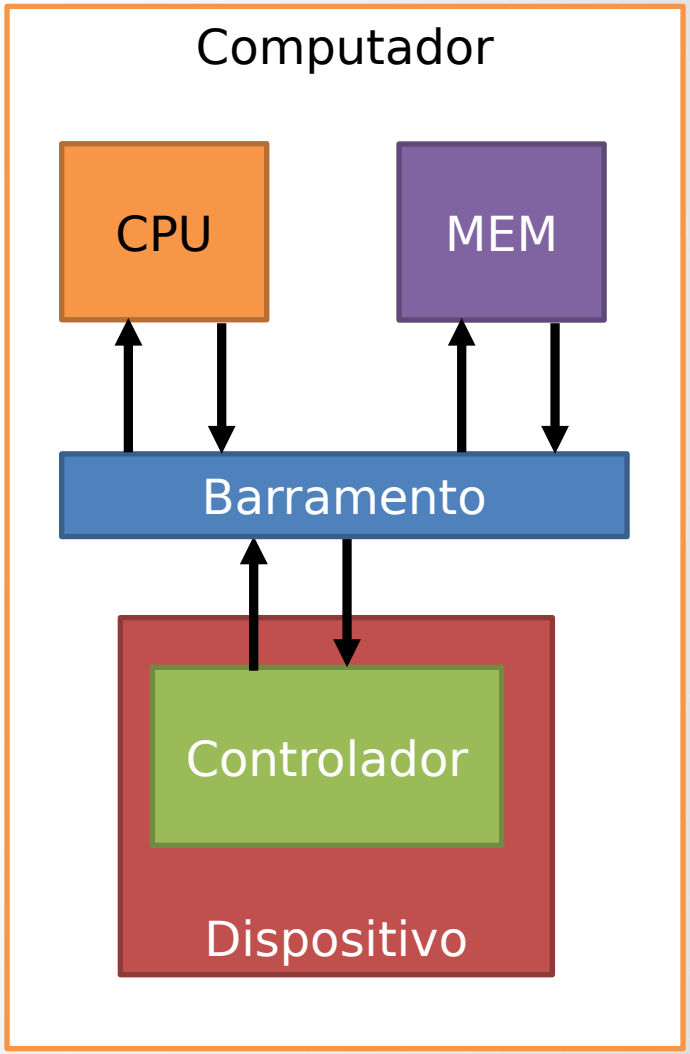
Princípios de Hardware de E/S

- Controladores de E/S
  - Cada controlador tem alguns **registradores** que são endereçados pela CPU a acessados diretamente
  - Uma possibilidade é usar **espaço de endereçamento especial para E/S**, sendo que cada controlador é designado para um intervalo.
  - SO executa entrada e saída “**escrevendo**” comandos nos registradores dos controladores.
  - Quando comando é aceito, CPU é usada para outro processo.

Princípios de Hardware de E/S

- Controladores de E/S
  - Quando requisição é realizada, controlador gera um sinal eletrônico que causa uma **interrupção**.
  - Sistema consulta vetor de interrupção para localizar rotina de tratamento.
  - CPU faz a transferencia da informação do buffer do controlador para a memória.

Princípios de Hardware de E/S

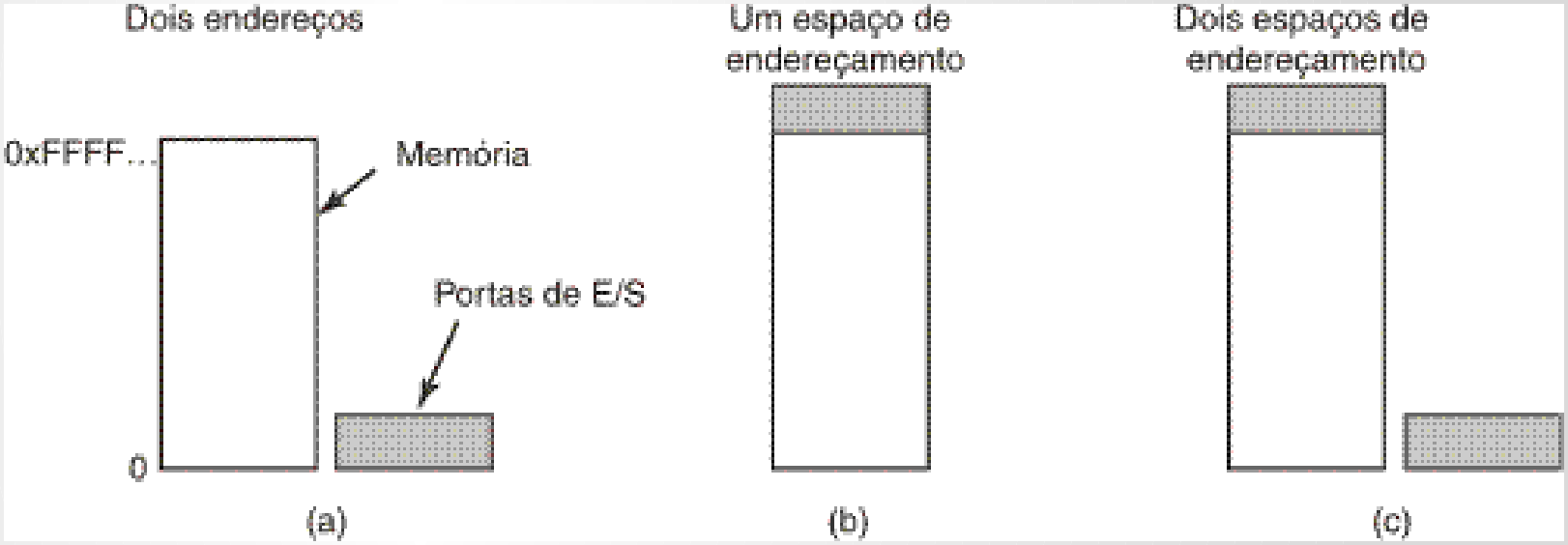


Princípios de Hardware de E/S

- Como se comunicar com os registradores dos controladores?
- Duas maneiras:
  1. Cada registrador é associado a uma **porta de E/S**. O conjunto delas forma o **espaço de portas de E/S.**
     - Somente o S.O pode acessá-las por meio de instruções privilegiadas
  2. **E/S Mapeada em memória**:
     - Cada registrador de controle é associado a um endereço de memória
- Existe ainda um mecanismo Híbrido:
  - Buffers de dados de E/S são mapeados na memória e as portas de E/S separadas para os registradores de controle

Princípios de Hardware de E/S

- E/S mapeada em Memória

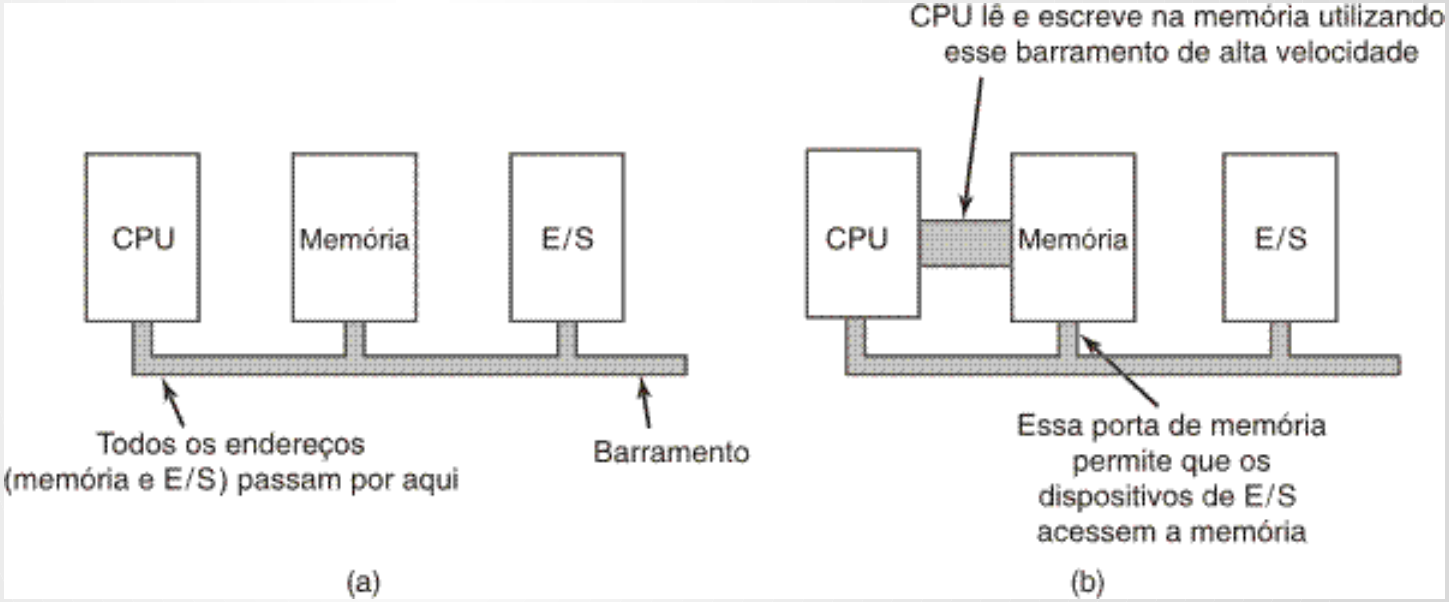


Princípios de Hardware de E/S

- E/S mapeada em memória
  - Vantagens:
    - Não necessita de **instruções especiais** para acessar os registradores de controle: tudo são variáveis na memória
    - Não necessita de um **mecanismo de proteção especial**. O acesso a região da memória com as informaç~eos de E/S pode ser protegida pelo sistema operacional omitindo-a do espaço de endereçamento dos outros usuários.
    - Instruções comuns podem fazer uso dos registradores de controle
  - Desvantagens:
    - Espaço de memória deve ser não-cacheável
    - Todos os dispositivos devem verificar o envio dos dados no barramento. Entretanto nem toda comunicação de memória passa pelo barramento compartilhado.

Princípios de Hardware de E/S

- E/S mapeada em memória

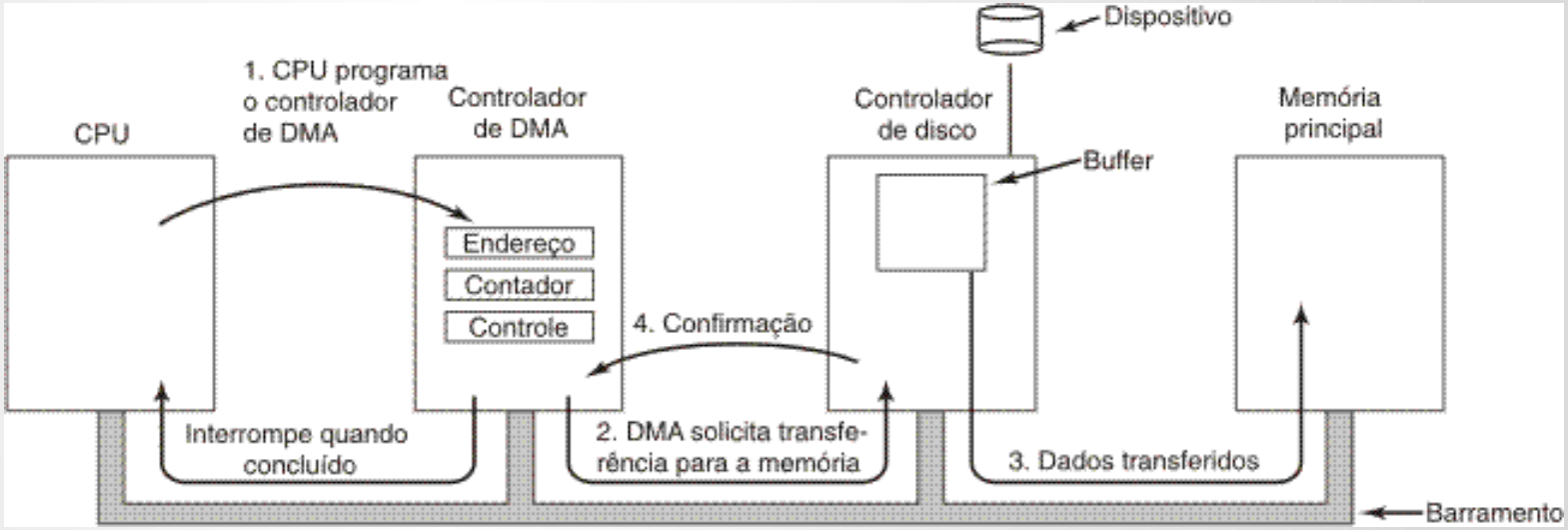


Princípios de Hardware de E/S

- Acesso Direto à Memória (DMA)
  - Exige um dispositivo físico que controla a transferência de dados
  - Programável pelo processador
  - **Word-at-a-time-mode** vs **burst-mode**
    - Roubo de ciclo: caso a CPU queira o barramento, ela é negada.
  - Uso de um buffer interno
    - Dá oportunidade para fazer verificação de erro (**checksum**)

Princípios de Hardware de E/S

- Controlador de DMA

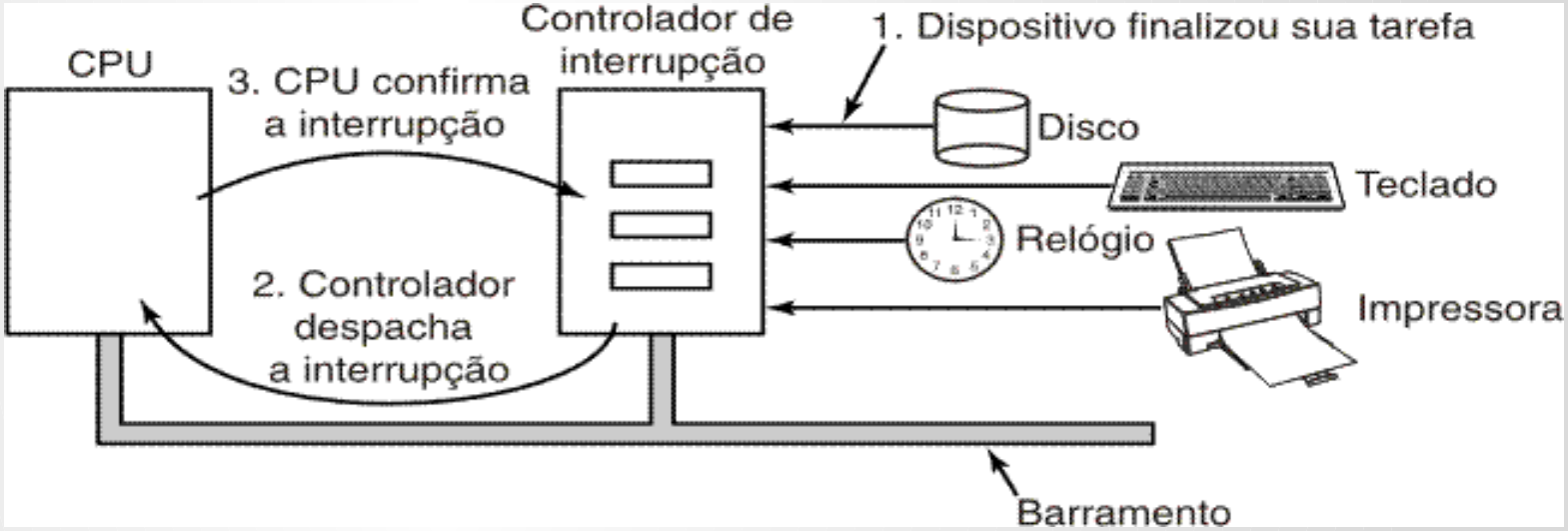


Princípios de Hardware de E/S

- Interrupções
  - Vetor de interrupções
  - Salvar o contexto
    - Salvar em registradores internos...
      - ...mas existe a possibilidade de uma segunda interrupção (independente) sobreescrever os dados
    - Salvar em pilha
      - De quem é a pilha? Do usuário?
      - O que acontece em uma falta de página?
  - **Precisas** vs. **Imprecisas**
    - Todas as instruções anteriores foram concluídas?
    - Como gerenciar isso?

Princípios de Hardware de E/S

- Interrupções



PRINCÍPIOS DE SOFTWARE DE E/S

Princípios de Software de E/S

- **Independência de dispositivo**
  - Programas podem acessar qualquer dispositivo de E/S sem especificar previamente qual (disquete, disco rígido ou CD-ROM)
- **Nomeação uniforme**
  - Nome de um arquivo ou dispositivo pode ser uma cadeia de caracteres ou um número inteiro que é independente do dispositivo
- **Tratamento de erro**
  - Trata o mais próximo possível do hardware

Princípios de Software de E/S

- **Transferências Síncronas vs. Assíncronas**
  - transferências bloqueantes vs. orientadas a interrupção
  - utilização de buffer para armazenamento tempoirário
  - dados provenientes de um dispositivo muitas vezes não podem ser armazenados diretamente em seu destino final
- **Dispositivos Compartilháveis vs. Dedicados**
  - discos são compartilháveis
  - unidades de fita não são

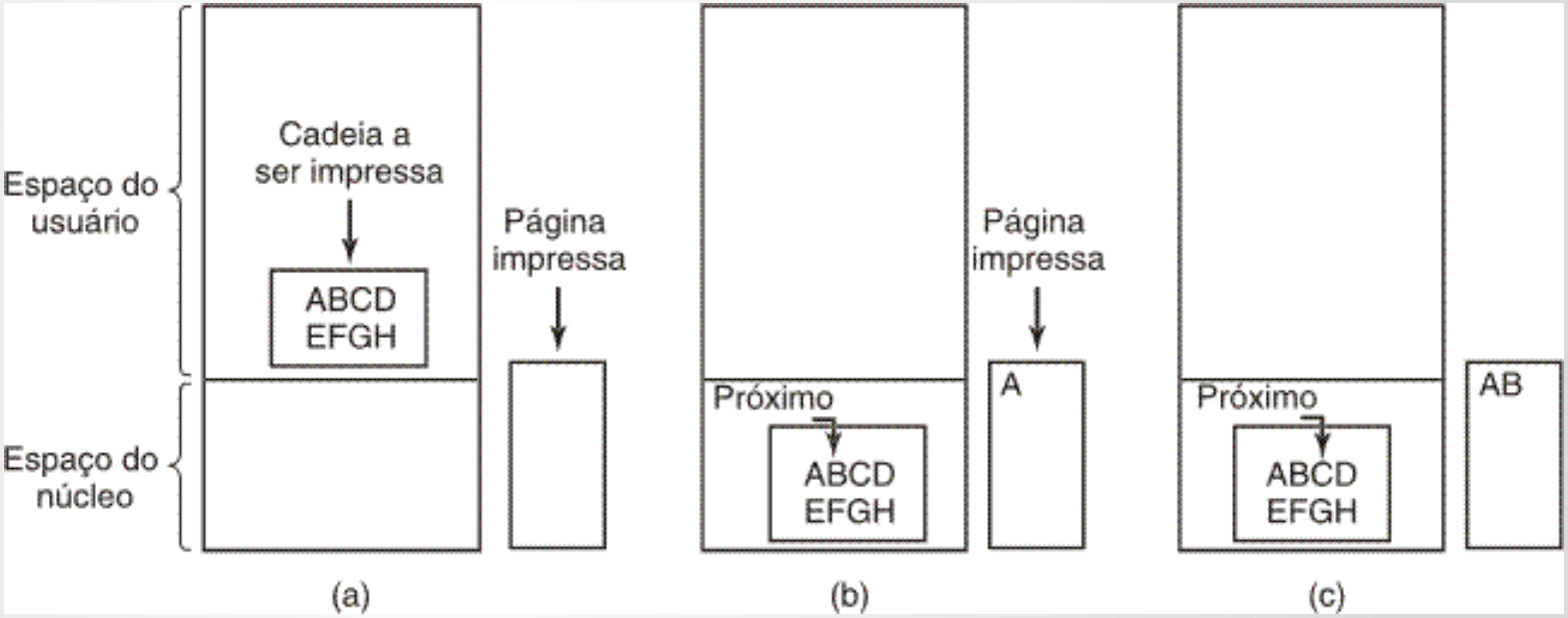
Princípios de Software de E/S

- E/S programada
  - Faz uso do processador para executar toda a transferencia de dados para/do dispositivo
  - Espera ocupada (busy-waiting)
  - As vezes não é um problema
    - Certos sistemas são monousuário ou tem uma carga de trabalho baixa

```
copy_from_user(buffer, p, cont);                /* p é o buffer do núcleo */
for (i=0; i < count; i++) {                     /* executa o laço para cada caractere */
    while (*printer_status_reg !=READY) ;       /* executa o laço até PRONTO */
    *printer_data_register = p[i];              /* envia um caractere para a saída */
}
return_to_user();
```

Princípios de Software de E/S

- E/S Programada



Princípios de Software de E/S

- E/S usando interrupção
  - Para permitir a exeução de outros processos durante a espera pelo dispositivo estar disponível, utliza-se interrupção

```
copy_from_user(buffer, p, count);
enable_interrupts( );
while (*printer_status_reg != READY) ;
*printer_data_register = p[0];
scheduler( );
```

(a)

```
if (count == 0) {
    unblock_user( );
} else {
    *printer_data_register = p[i];
    count = count – 1;
    i = i + 1;
}
acknowledge_interrupt( );
return_from_interrupt( );
```

(b)

Princípios de Software de E/S

- E/S usando DMA
  - O processador programa o DMA estabelecendo endereços iniciais de cópia dos dados (buffer)

```
copy_from_user(buffer, p, count);
set_up_DMA_controller( );
scheduler( );
```

(a)

```
acknowledge_interrupt( );
unblock_user( );
return_from_interrupt( );
```

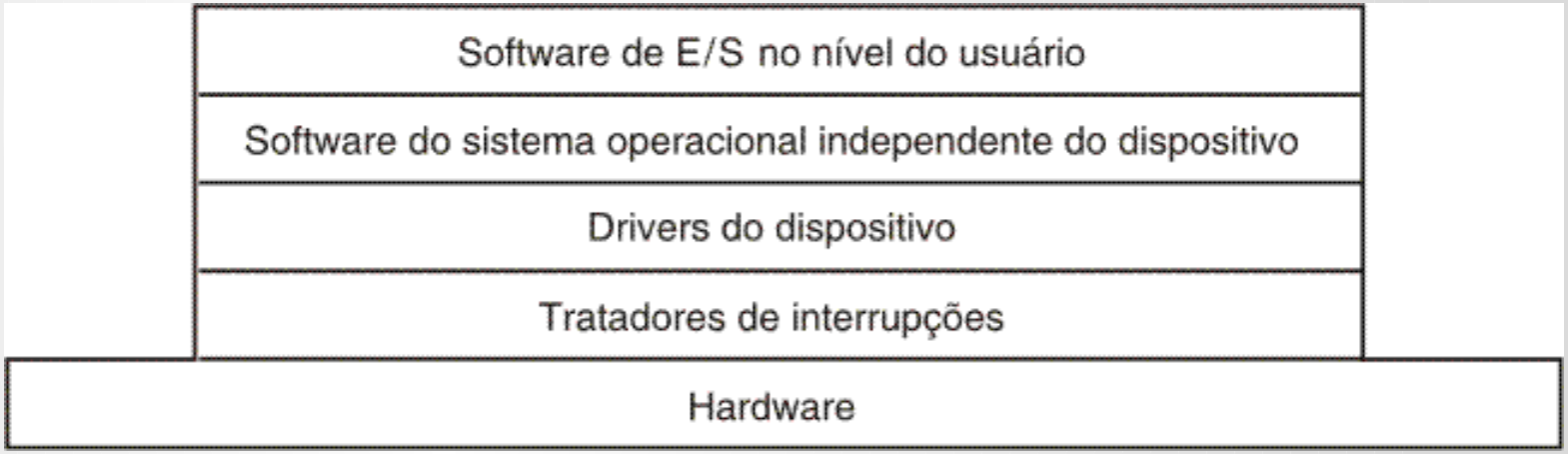
(b)

CAMADAS DE SOFTWARE DE E/S

Camadas de Software de E/S

- Objetivos
  - Independência do tipo de dispositivo
    - “sort < input > output” deveria funcionar sempre
  - Uniformidade da escolha de nomes
    - Independente do dispositivo
    - E.g. em Unix podemos montar dispositivos de bloco em qualquer diretório
  - Detecção de recuperação de erros
    - Deve ser feita o mais próximo possível do hardware possível
    - Maior parte dos erros pode ser corrigido por repetição
  - Bloqueio vs. Interrupção
    - No nível de usuário geralmente bloqueio – mais fácil de codificar
    - No nível do SO geralmente interrupção , pois permite utilização dos recursos do sistema enquanto E/S é realizada

Camadas de Software de E/S



Camadas de Software de E/S

- Quatro camadas
  1. Processamento de interrupção
  2. Drivers de dispositivo
  3. Software de E/S independente de dispositivo
  4. Software do usuário

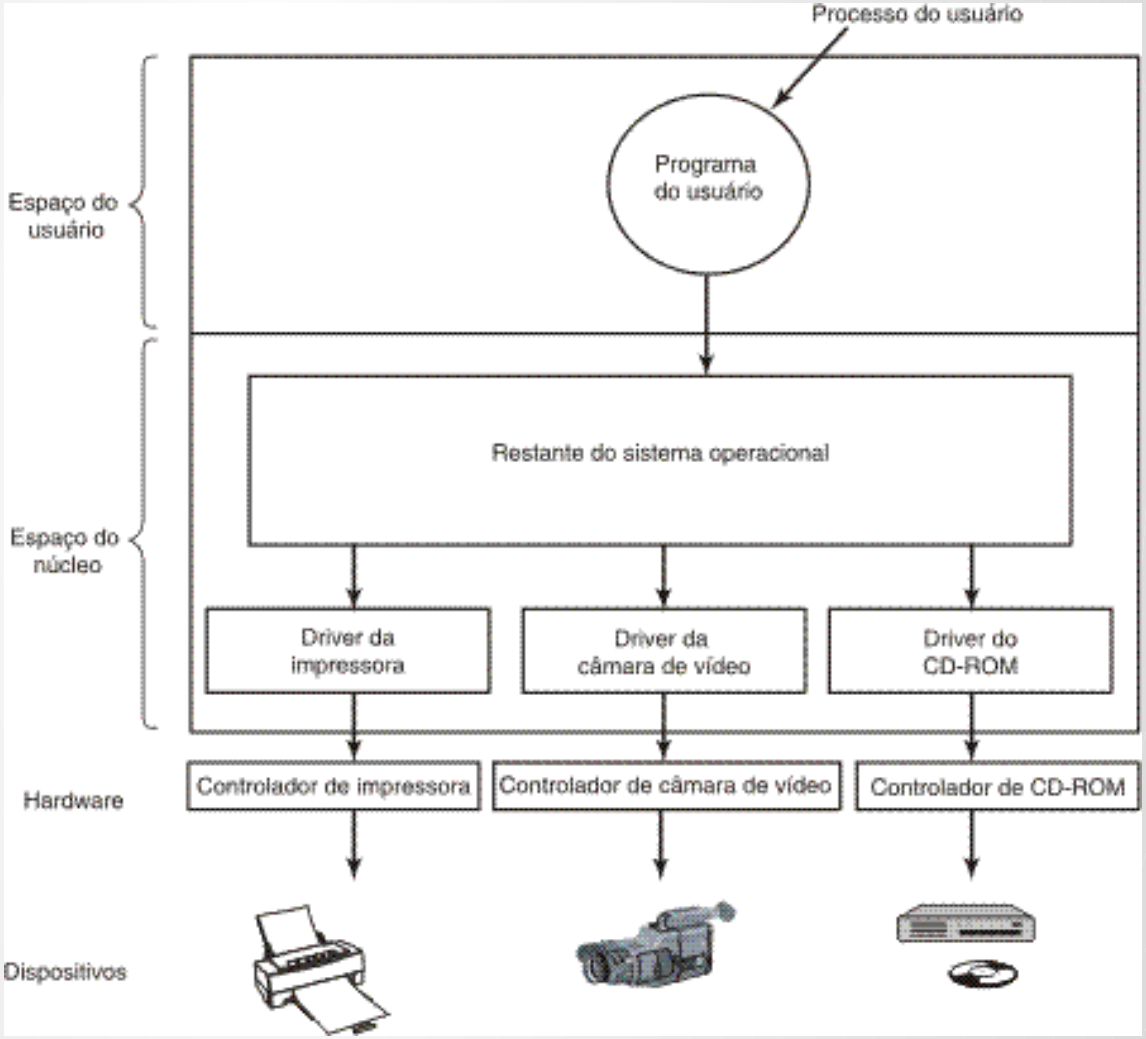
Processamento de Interrupção

- Interrupções eliminam a ilusão de multiprocessamento
- Objetivo: esconder interrupções o mais internamente possível ao sistema
- Em Minix, processos são bloqueados quando comando de E/S emitido e interrupção é esperada (mesmo drivers)
- Quando interrupção ocorre o processador de interrupção faz o necessário para desbloquear processo
  - V() em semáforo, envio de mensagem (Minix), etc.

Drivers de Dispositivos

- Um para cada tipo de dispositivo (ou classe de dispositivos semelhantes, ex. SATA)
- Encarregado de traduzir comandos emitidos pelo software independente de dispositivo para particularidades do controlador
  - Inicio e fim de buffer, tamanho da tansferencia, etc.
- Encapsula conhecimento sobre
  - Número e função dos registradores de um controlador
  - Particularidades na organização do dispositivo
    - Linear, cilindros+trilhas+setores,
    - cabeças,
    - movimento do braço do disco,
    - fatores de interleaving,
    - atrasos mecânicos

Drivers de Dispositivos



Camadas de Software de E/S

- Exemplo: Leitura do HDD

1. Bloco N é requisitado
2. Se driver está livre, pedido é aceito, senão requisição e colocada em fila de espera
3. Requisição é traduzida em termos concretos:
    1. verificar onde estão setores correspondentes ao bloco pedido,
    2. onde está braço do disco,
    3. se motor do braço está funcionando,
    4. se o disco está girando
4. Decide comandos a serem emitidos (iniciar rotação, mover braço, etc.
5. Emite comandos, escrevendo nos registradores do controlador (um de cada vez ou lista ligada, dependendo do controlador)
6. Driver bloqueia esperando resultado
    - Quando operação termina sem demora (ex. Memória gráfica), driver não precisa bloquear
7. Após operação concluída, verifica erros, e retorna resultado ou trata erro

Software Independente de Dispositivo

- A maior parte do software de E/S é independente de dispositivo
- Cuidado: ainda temos CLASSES de E/S: bloco e caracter
- Divisão exata de limites entre software independente de dispositivo e drivers depende do sistema
  - em alguns sistemas, driver faz funções poderiam ser independentes, oferecendo visão mais abstrata

Software Independente de Dispositivo

- Funções básicas
  1. Interface uniforme para drivers
  2. Administração de nomes de dispositivos
     - Mapeia nomes simbólicos para dispositivos reais
     - Em Minix I-node para arquivo especial com “major device number usado para localizar driver e “minor device number” para passado para driver como parâmetro
  3. Proteção dos dispositivos
     - Como prevenir acesso indevido?
     - Minix/UNIX usa proteção do sistema de arquivos
  4. Possibilitar blocos de tamanho padrão
     - Dispositivos diferentes podem ter tamanhos de setores distintos
     - Camadas mais altas vêem apenas “dispositivo virtual” padrão.

Software Independente de Dispositivo

- Funções básicas (cont.)
  1. “Buffering”
     - Bloco: Hardware lê/grava blocos grandes, usuário pode ler/gravar até um byte
     - Char: usuários podem escrever mais rápido que dispositivo pode aceitar/ teclado pode chegar antes do uso.
  2. Cuidar da administração do espaço alocado em dispositivos de bloco
  3. Reservar e liberar dispositivos dedicados
     - Fitas magnéticas, gravador de backup, dvd, ...
  4. Relato de erros
     1. Maior parte feita pelos drivers, já que erros em sua maioria são específicos para cada disponsitivo
     2. Porém, ex: quando bloco não podem mais ser lido?

Software no nível do usuário

- Rotinas de E/S em Bibliotecas
  - Printf, scanf, fprintf, etc
- Comandos de leitura/escrita em linguagens
- Colocam parâmetros nos locais corretos para chamadas de sistema
- Rotinas que constroem E/S a partir de especificações
  - Atoi, printf
- Outros programas
  - Spooling
    - Spooling directory + daemon; daemon é o único com acesso à impressora
  - Email, acesso à redes, etc.

Proxima aula

- Exercícios!

Referências

- TANENBAUM, Andrew S.. **Sistemas operacionais modernos**. 3. ed. São Paulo: Prentice Hall, 2009. 653 p. ISBN: 9788576052371.
  - **Capítulo 5**

- OLIVEIRA, Rômulo Silva de; CARISSIMI, Alexandre da Silva; TOSCANI, Simão Sirineo. **Sistemas operacionais**. 4. ed. Porto Alegre: Bookman, 2010. ISBN: 9788577805211.
  - **Capitulo 8**

- SILBERCHATZ, A.; Galvin, P.; Gagne, G.; **Fundamentos de Sistemas Operacionais**, LTC, 2015. ISBN: 9788521629399
  - **Capítulo 9**